**Московский государственный технический университет им.Н.Э.Баумана**

**кафедра "Системы обработки информации и управления"**



**Веб-сервис по продажам автомобилей**

**Отчет**

к лабораторной работе №8

Инструктор：Мария Валерьевна, Elizaveta Eliseeva, Masha

Email:2623859464@qq.com

Студент: Ван Чаочао

группа ИУ5И-12М

2021/12/30

### Цель работы:

Изучить методы подготовки и проведения тестирования и получить навыки

создания и выполнения тестов для приложений и их компонентов.

### Порядок и время проведения работы:

Время проведения работы 4 часа. Работа проводится в компьютерном зале и

выполняется по индивидуальной теме, выданной преподавателем.

### Задание

1) Модульное тестирование (**базовое**):

− Открыть исходный код тестируемого приложения.

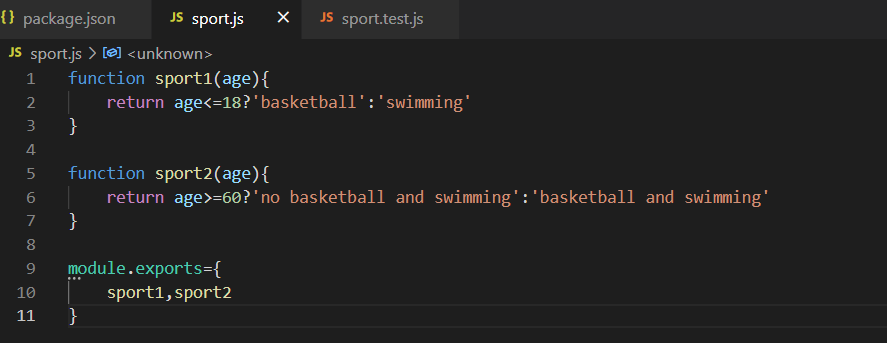
− Добавить Unit-тест для одной из функций.

− Запустить тест и просмотреть результаты.

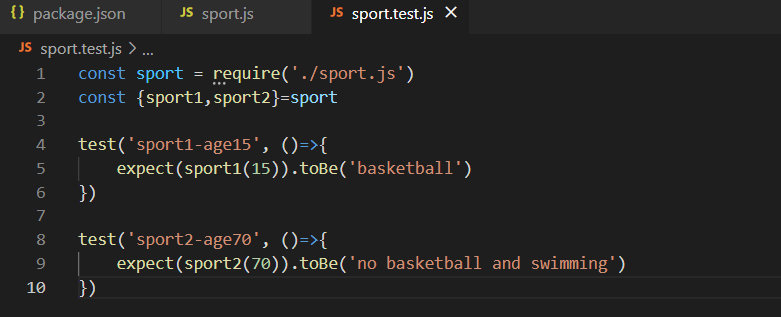
− Создать несколько разных тестов (для проверки значений и перехвата

исключений).

**Sport.js**



**Sport.test.js**



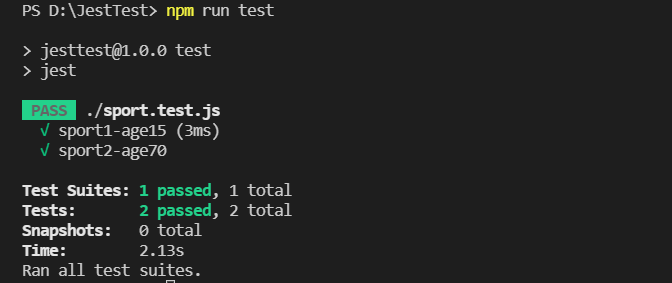
**Результаты тестирования:**

Функция тест возвращает либо успешное сведения о результатах выполнения тестов.

Если тесты выполнены успешно и полученный результат равен ожидаемому то возвращается

PASS, в противно случае возвращается ERROR. Результаты тестирование приведены ниже.

**npm run test**

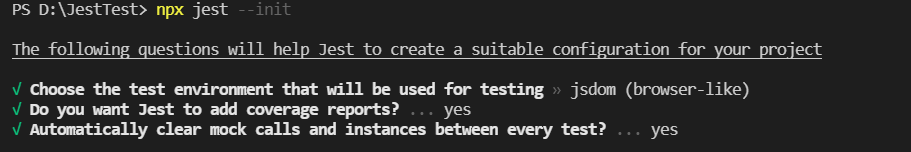


2) Покрытие кода тестами (**базовое**):

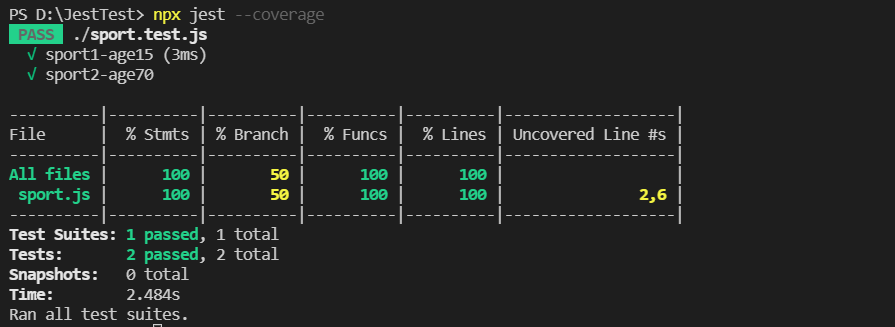
− Установить параметры сбора статистики покрытия кода.

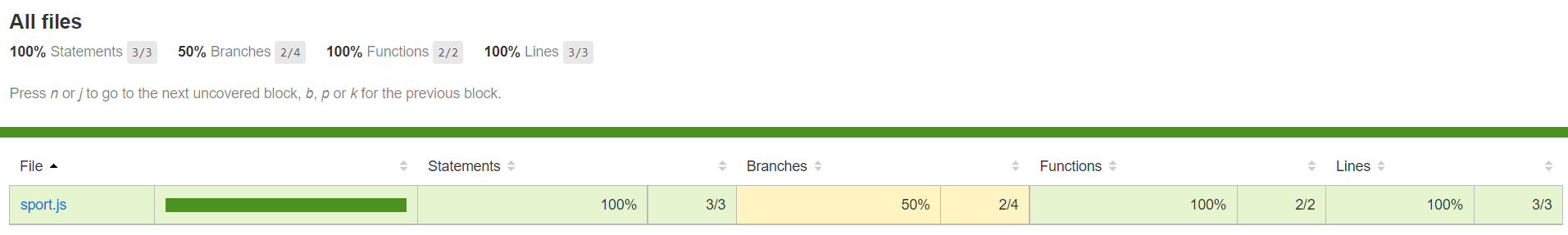
− Повторить модульные тесты и просмотреть данные о покрытии кода.

**npx jest --init(生成初始化配置)**



**npx jest --coverage**





3) Веб-тесты производительности (**расширенное**) :

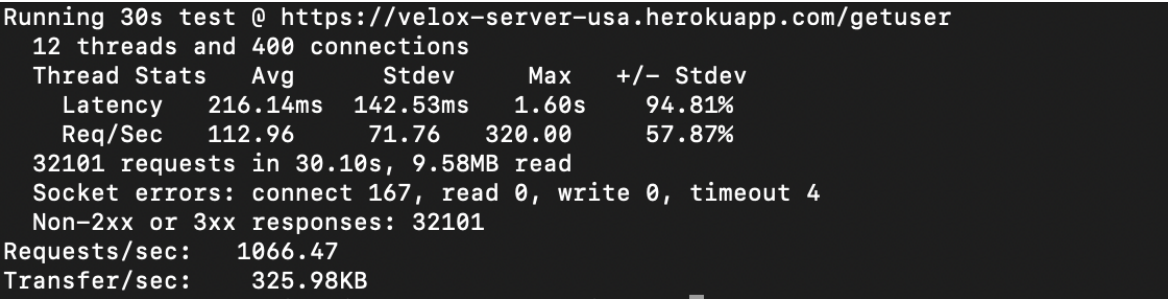
− Создать тестовый проект по веб-тестам производительности (для своего

сайта или любого стандартного). При этом записать сценарий работы с

сайтом.

− Настроить параметры нагрузки (частота запросов и т.д.)

− Выполнить тест и просмотреть результаты.

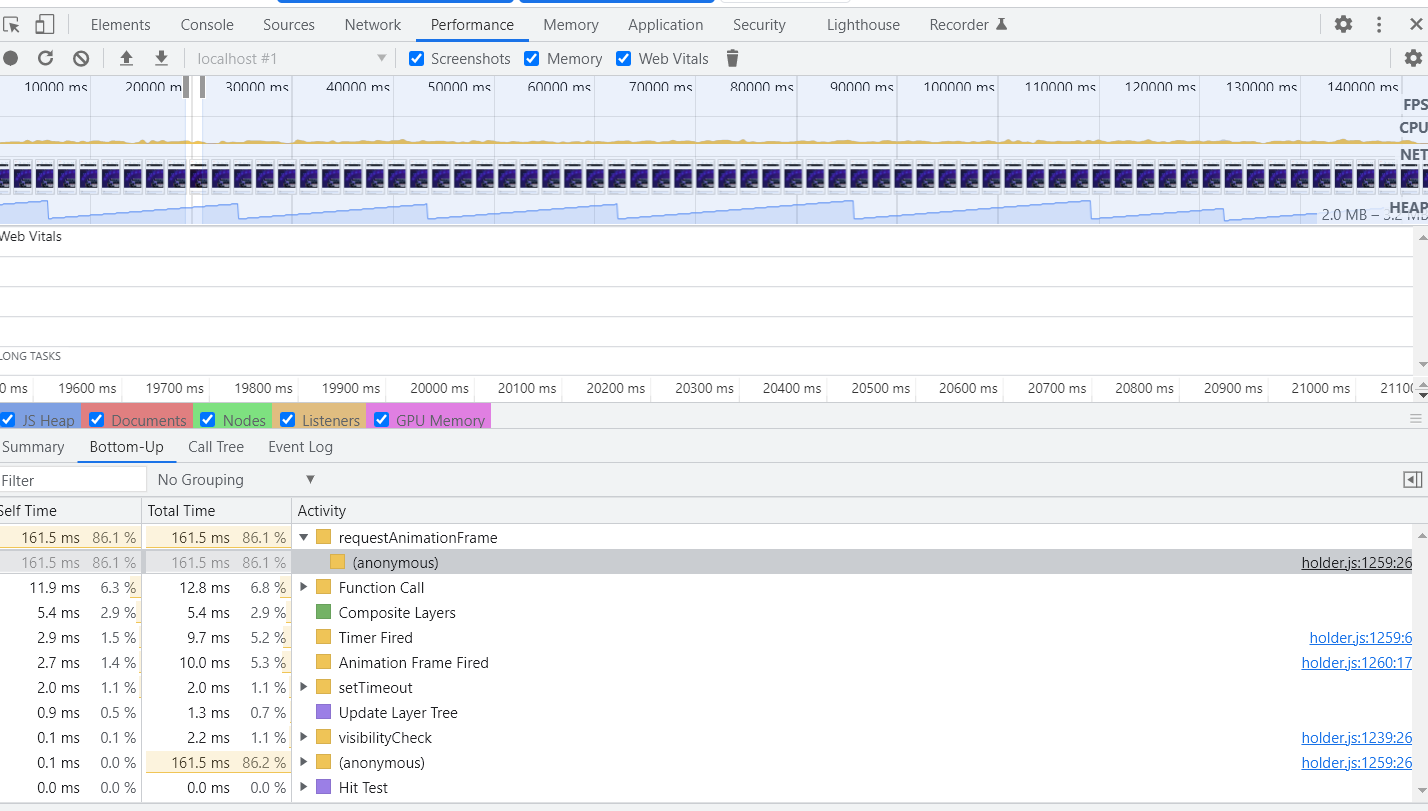


4) Нагрузочное тестирование (**расширенное)**:

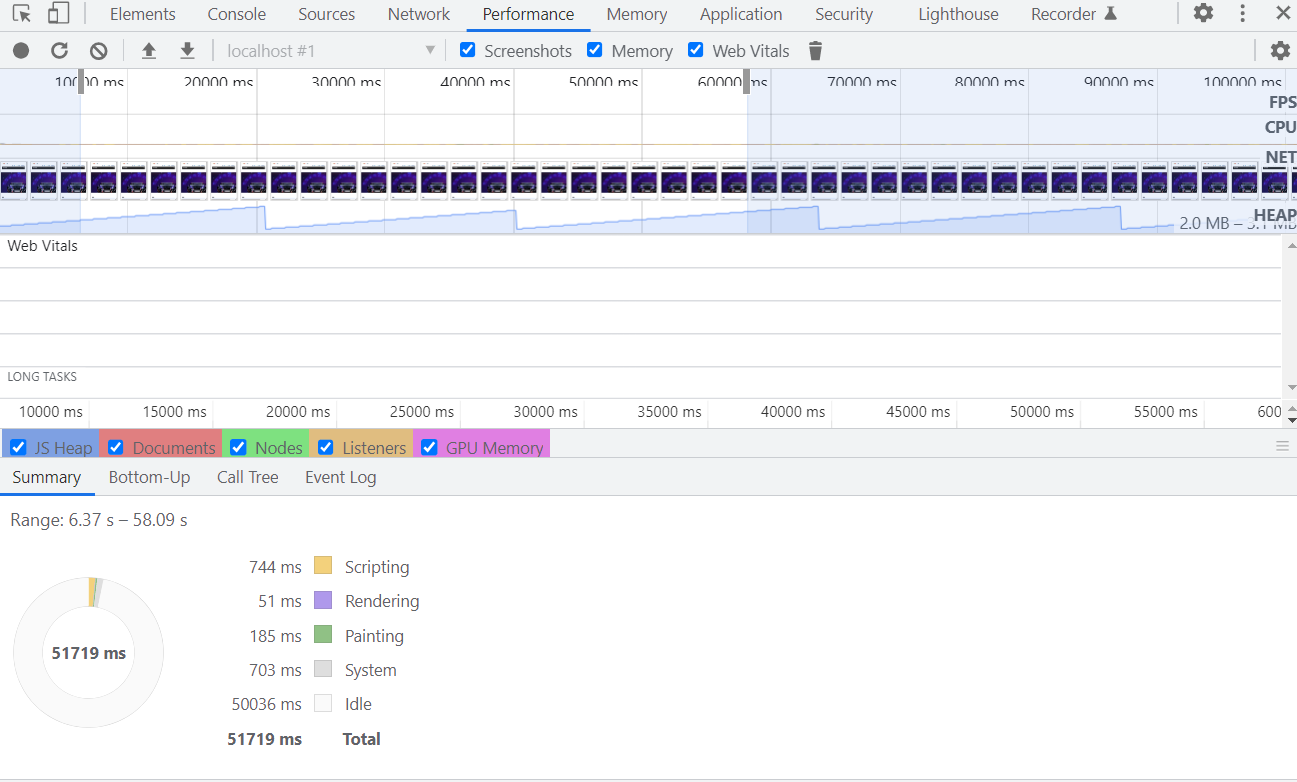
− Для тестируемого приложения провести профайлинг (оценку

производительности).

− Выполнить тест и просмотреть результаты.

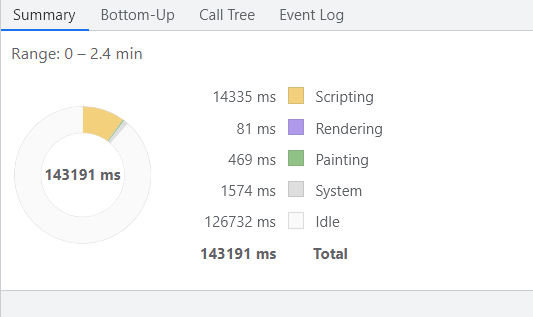


**Общий график**



**Запуск тестов**

Результаты



Рассмотрим полученные результаты.

Выполнение и загрузка скрипта (Scripting) страницы заняло 14335 мс

Рендеринг, а именно отрисовка страницы (Rendering) страницы занял 81 мс

Погрузка css - стилей страницы (Painting) страницы занял 469 мс

Погрузка системных браузерных настроек (System) страницы занял 1574 мс

Время простоя системы после выполнения всех процессов до окончания нагрузочного

тестирования приложения (Idle) страницы занял 126732 мс

Исходя из результатов полученных нами мы можем судить о довольно быстром приложении.

5) Тесты GUI (**дополнительно**):

− Создать тестовый проект закодированного тестирования пользовательского

интерфейса.

− Наполнить тест.

− Выполнить тест и просмотреть результаты.

**Источники:**

1. Орлов С.А., Цилькер Б.Я. Технологии разработки программного обеспечения. - СПб.:

Питер. - 2012г

2. Виноградова М.В., Белоусова В.И. Унифицированный процесс разработки программного

обеспечения: учебное пособие / Виноградова М.В., Белоусова В.И. – М.: МГТУ им.Н.Э.

Баумана. – 2015 г. – 82 с. - Режим доступа:

http://ebooks.bmstu.ru/catalog/193/book1303.html (дата обращения: 17.12.2017). — ISBN:

978-5-7038-4265-2

3. Мищевский Г. Тестирование. Фундаментальная теория. - 2013. URL:

https://habr.com/ru/post/279535/